Graph Data Exchange with Target Constraints

Iovka Boneva, Angela Bonifati, Radu Ciucanu

Graph Data Exchange with Target Constraints

Iovka Boneva Angela Bonifati Radu Ciucanu
University of Lille & INRIA, France
{iovka.boneva, angela.bonifati, radu.ciucanu}@inria.fr

ABSTRACT

Data exchange is the problem of translating data structured under a source schema according to a target schema and a set of source-to-target constraints known as schema mappings. In this paper, we investigate the problem of data exchange in a heterogeneous setting, where the source is a relational database, the target is a graph database, and the schema mappings are defined across them. We study the classical problems considered in data exchange, namely the existence of solutions and query answering. We show that both problems are intractable in the presence of target constraints, already under significant restrictions.

1. INTRODUCTION

Data exchange is the problem of translating data structured under a source schema according to a target schema and a set of source-to-target constraints [11]. Such a problem has been studied in settings where both the source and target schemas belong to the same data model, in particular relational and nested relational [15, 11], XML [3], or graph [5]. Settings in which the source and the target schema are of heterogeneous data models have not been considered so far, apart from combinations of relational and nested relational schemas in schema mapping tools [15, 13].

In this paper, we focus on the problem of exchanging data between relational sources and graph-shaped target databases, which might occur in several interoperability scenarios in the Semantic Web, such as ontology-based data access [14] and direct mappings [16]. Motivations to map relational data to graphs abound, due to the far majority of data residing in relational databases and the need of integrating large amounts of linked data.

We express the relationships between the source and the target via *schema mappings* [15, 11, 7] i.e., logical assertions between two conjunctive queries, one on the source and the other on the target. Schema mappings between graph databases have already been introduced in [5] and we adopt their syntax for expressing the consequents of relational-to-graph schema mapping assertions. We point out that the setting without target constraints directly follows from the results in [5] on graph-to-graph data exchange. Furthermore, motivated by the fact that target constraints have been largely investigated within relational data exchange but so far disregarded for graph data exchange [5], we add them to our setting.

In particular, we focus on two fundamental problems of interest: *existence of solutions* (i.e., given a source schema and an instance of it, a target schema, a set of source-to-target constraints, and a set of target constraints, decide whether there exists an instance of the target schema satisfying all given constraints) and *query answering* (i.e., computing the answers that hold for all solutions).

Our *main contributions* are the following:

- We show that in the presence of *target equality-generating dependencies* [6], both existence of solutions and query answering are intractable (NP-hard and coNP-hard, respectively). This holds even under significant restrictions.
- We relax the notion of target constraints by introducing $sameAs$[1] target constraints, inspired by RDF[2]. We show that the existence of solutions becomes tractable while query answering is intractable (coNP-hard) for the same restrictions as for the previous case.
- We show that the notion of graph patterns [4], employed for graph data exchange [5] as *universal representatives* of all solutions, cannot be used as such when adding target constraints.

We point out that our hardness results stand in terms of *query complexity* since in the proofs we have used a fixed source schema and instance, while the target schema and the mappings are part of the input.

We also point out that none of our results is specific to the relational-to-graph setting, and hold in any setting where the target is a graph and the source is an arbitrary data model projecting on relational tuples. The source data can then be either XML, graph-shaped, or any other complex format as long as the left-hand sides of mappings extract relational tuples from it. Indeed, we shall pinpoint that the constraints on the target graph are solely responsible for the intractability results. Nevertheless, in the remainder of the paper, we focus on a relational data source for ease of presentation.

[1] `http://www.w3.org/wiki/WebSchemas/sameAs`
[2] `http://www.w3.org/RDF/`

Organization. In Section 2, we define our problem setting. In Section 3, we illustrate particular cases that can be solved using techniques from relational and graph data exchange. In Section 4, we characterize the complexity of the problems of interest. In Section 5, we present the challenges of defining and querying universal solutions. We discuss conclusions and future work in Section 6.

2. PROBLEM SETTING

Let us assume a countably infinite set of *constants* $\mathcal{V}$ that we use both as *domain* of relational databases and as *node identifiers* (or simply *node ids*) of graph databases.

Source schemas and queries. A *source schema* $\mathcal{R}$ is a finite collection of relational symbols. Each relational symbol has an *arity* that is a positive integer. An *instance* I of $\mathcal{R}$ is a function associating to each relational symbol R from $\mathcal{R}$ a set of tuples over $\mathcal{V}$ having the same arity as R. We abuse notation and use R to denote both relational symbol and its instance. A *source query* is a conjunction of atoms over $\mathcal{R}$ that uses only variables.

Target schemas and queries. A *target schema* Σ is a finite alphabet. An *instance* over Σ is a *directed, edge-labeled graph* $G = (V, E)$, where $V \subseteq \mathcal{V}$ is a finite set of node ids and $E \subseteq V \times \Sigma \times V$ is a finite set of edges. A *nested regular expression* (*NRE*) is an expression of the following grammar:

$$r := \varepsilon \mid a \ (a \in \Sigma) \mid a^- \ (a \in \Sigma) \mid r + r \mid r \cdot r \mid r^* \mid [r],$$

where "$+$" stands for disjunction, "$\cdot$" for concatenation, "$*$" for Kleene star, "$^-$" for traversing edges backwards, and "$[\,]$" for nesting. An NRE r defines a binary relation over graph nodes: $[\![r]\!]_G$ is the set of pairs of nodes in G s.t. there exists a path defined by r between the two nodes. We consider the semantics of NREs as in [5]. A *target query* is a *conjunction of nested regular expressions* (*CNRE*) using variables only. We illustrate CNREs in Example 2.2.

Schema mappings. A schema mapping is a set of *source-to-target tuple-generating dependencies* [6] (or simply *s-t tgds*) i.e., a set of formulas of the form

$$\forall \overline{x}.\ (\phi_{\mathcal{R}}(\overline{x}) \rightarrow \exists \overline{y}.\ \psi_{\Sigma}(\overline{x}, \overline{y})),$$

where $\phi_{\mathcal{R}}(\overline{x})$ is query over $\mathcal{R}$ and $\psi_{\Sigma}(\overline{x}, \overline{y})$ is a query over Σ. By $\overline{x}$ and $\overline{y}$ we denote vectors of variables. Moreover, all variables in $\overline{x}$ appear free in $\phi_{\mathcal{R}}(\overline{x})$, all variables in $\overline{y}$ appear free in $\psi_{\Sigma}(\overline{x}, \overline{y})$, and the variables in $\overline{x}$ that appear in $\psi_{\Sigma}(\overline{x}, \overline{y})$ are free.

Target constraints. We consider two well-known types of target constraints:

- *target equality-generating dependencies* [6] (or simply *egds*) i.e., $\forall \overline{x}.\ (\psi_{\Sigma}(\overline{x}) \rightarrow (x_1 = x_2))$,
- *target tuple-generating dependencies* [6] (or simply *target tgds*) i.e., $\forall \overline{x}.\ (\phi_{\Sigma}(\overline{x}) \rightarrow \exists \overline{y}.\ \psi_{\Sigma}(\overline{x}, \overline{y}))$.

In the aforementioned definitions, $\phi_{\Sigma}(\overline{x})$ and $\psi_{\Sigma}(\overline{x})$ are CNREs over Σ, and x_1 and x_2 are among the variables in $\overline{x}$. Moreover, we introduce *sameAs target constraints* that are a special case of target tgds i.e.,

$$\forall \overline{x}.\ (\psi_{\Sigma}(\overline{x}) \rightarrow (x_1, \mathit{sameAs}, x_2)).$$

In the sequel, we omit w.l.o.g. the universal quantifiers in front of a formula.

Definition 2.1 *A* (relational-to-graph) data exchange setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ *consists of a relational source schema* $\mathcal{R}$*, a graph target schema* Σ*, a set* $\mathcal{M}_{st}$ *of s-t tgds, and a set* $\mathcal{M}_t$ *of target constraints.*

Solutions. Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$, an instance I of $\mathcal{R}$ and a graph database G over Σ, we say that G is a *solution* for I under Ω if (I, G) satisfies $\mathcal{M}_{st}$ and G satisfies $\mathcal{M}_t$. We denote the set of all solutions by $\mathit{Sol}_{\Omega}(I)$. Usually, in relational data exchange, one aims at finding the universal solutions, from which there exist homomorphisms to all solutions [11]. This notion has been redefined for graph data exchange as universal representatives captured with graph patterns [5] that we discuss in detail in Section 3.2.

Example 2.2 Take a source schema $\mathcal{R}$ consisting of two relations: *Flight* storing information about flights that may have intermediate stops between the source and destination cities, and *Hotel* storing information about the hotels in which the passengers of such flights have stopped. Moreover, take the following instance I:

Flight		
flight_id	*src*	*dest*
01	c_1	c_2
02	c_3	c_2

Hotel	
flight_id	*hotel_id*
01	hx
01	hy
02	hx

Take a target schema consisting of the alphabet $\Sigma = \{f, h\}$. The edges labeled by f indicate a direct flight between two cities while the edges labeled by h indicate that a city has a hotel. Moreover, consider the following s-t tgd that basically requires that for each hotel where the passengers of a flight have stopped, there exists a city where the respective hotel is situated, and there exist flights from *src* to such city and from such city to *dest*:

$$\mathcal{M}_{st}:\ \mathit{Flight}(x_1, x_2, x_3) \wedge \mathit{Hotel}(x_1, x_4) \rightarrow \exists y.\ (x_2, (f \cdot f^*), y) \wedge (y, h, x_4) \wedge (y, (f \cdot f^*), x_3).$$

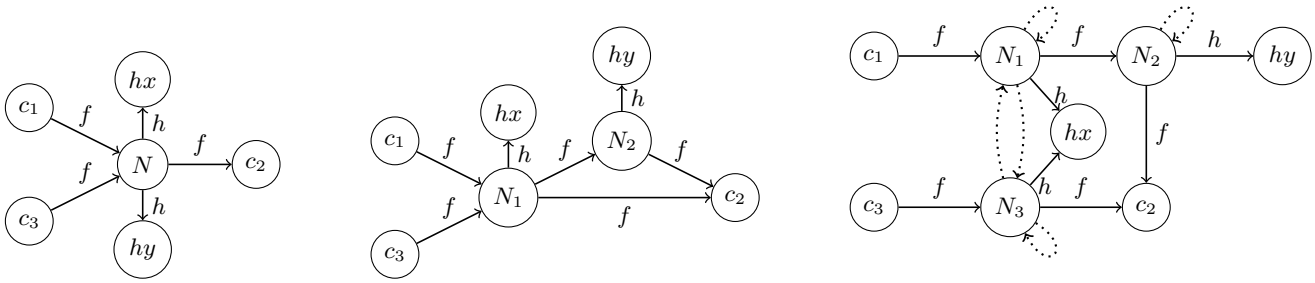
Notice that $\mathcal{M}_{st}$ uses a CNRE on its right hand side. Then, a natural constraint is that a hotel is situated in exactly one city, which can be captured either by the egd $\mathcal{M}_t$ or by the *sameAs* constraint $\mathcal{M}'_t$:

$$\mathcal{M}_t:\ (x_1, h, x_3) \wedge (x_2, h, x_3) \rightarrow (x_1 = x_2),$$
$$\mathcal{M}'_t:\ (x_1, h, x_3) \wedge (x_2, h, x_3) \rightarrow (x_1, \mathit{sameAs}, x_2).$$

The two ways of expressing the aforementioned target constraint yield two different settings $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ and $\Omega' = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}'_t)$, respectively. We illustrate in Figure 1 solutions for I under these two settings: the graphs G_1 and G_2 are solutions under Ω, while G_3 is a solution under Ω'. ▫

Problems of interest. We are interested in studying the following two problems:

1. *Existence of solutions.* Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ and an instance I of $\mathcal{R}$, decide whether



(a) G_1. (b) G_2. (c) G_3.

Figure 1: Solutions from Example 2.2. The dotted edges are labeled by *sameAs*.

there exists a solution for I under Ω. Additionally, we are interested in finding in our heterogeneous setting a mechanism similar to universal solutions [11] or universal representatives [5].

2. *Query answering.* Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$, an instance I of $\mathcal{R}$, and a query Q over Σ, we are interested in the *certain answers of Q w.r.t. I under Ω*, denoted $cert_\Omega(Q, I)$, which are the answers that hold for all solutions i.e., the set $\bigcap\{[\![Q]\!]_G \mid G \in Sol_\Omega(I)\}$ (where by $[\![Q]\!]_G$ we denote the set of tuples of nodes of G selected by the query Q). The query answering problem consists of deciding whether a given tuple of constants belongs to $cert_\Omega(Q, I)$ or not.

Example 2.2 (continued). Take the above instance I of the relations *Flight* and *Hotel*, and the above setting Ω. Then, take the query

$$Q = (x_1, f \cdot f^*[h] \cdot f^- \cdot (f^-)^*, x_2).$$

Intuitively, this query selects the pairs of nodes (x_1, x_2) from which the same hotel can be reached, or in other words, one can fly (possibly with connections) from the city x_1 to another city that has a hotel and an ingoing flight (possibly with connections) whose origin x_2 we want to select. Recall that the graphs G_1 and G_2 are both solutions for I under Ω. On these two graphs, the query Q selects as follows:

$$\begin{aligned}[\![Q]\!]_{G_1} &= \{(c_1, c_1), (c_1, c_3), (c_3, c_1), (c_3, c_3)\},\\ [\![Q]\!]_{G_2} &= \{(c_1, c_1), (c_1, c_3), (c_3, c_1), (c_3, c_3),\\ &\quad (c_1, N_1), (c_3, N_1), (N_1, c_1), (N_1, c_3), (N_1, N_1)\}.\end{aligned}$$

Notice that only four pairs of nodes are common to these answer sets for the two considered graphs. Also notice that these four pairs of nodes are in fact the certain answers of Q w.r.t. I under Ω:

$$cert_\Omega(Q, I) = \{(c_1, c_1), (c_1, c_3), (c_3, c_1), (c_3, c_3)\}.$$

On the other hand, notice that if we want to pose the same query Q under the other aforementioned example of setting (i.e., Ω'), we obtain a different set of certain answers: $cert_{\Omega'}(Q, I) = \{(c_1, c_1), (c_3, c_3)\}$. Intuitively, this happens because the egds from the setting Ω ensure that in all of its possible solutions the nodes having the same hotel have been merged. In the second setting, this natural requirement has been encoded using a *sameAs* constraint, which is not exploited by the query Q, hence some of the certain answers of Q under Ω are no longer certain under Ω'. ▫

3. BACKGROUND

In this section, we show that in two particular cases of our problem setting existing techniques from relational and graph data exchange can be applied (Section 3.1 and Section 3.2, respectively). This does not happen in the general case, as we show in the next section.

3.1 Relational data exchange

If we consider s-t tgds having on the right hand side conjunctions of NREs of the form a (with $a \in \Sigma$), our problem setting reduces to a particular case of relational data exchange and the techniques from relational data exchange can be naturally applied. In particular, we can see the target schema as a set of binary relational symbols (one for each symbol of the target alphabet) and the chase [11] returns a universal solution that can be essentially seen as a graph since it consists of a set of binary relations.

Example 3.1 Take the schemas $\mathcal{R}$ and Σ, the instance I, and the egds $\mathcal{M}_t$ from Example 2.2. Since we consider only NREs of the form a (with $a \in \Sigma$), we cannot express the same $\mathcal{M}_{st}$ as in Example 2.2. However, we can express the following:

$$\begin{aligned}\mathcal{M}'_{st}: \ & \mathit{Flight}(x_1, x_2, x_3) \wedge \mathit{Hotel}(x_1, x_4) \rightarrow\\ & \exists y.\ (x_2, f, y) \wedge (y, h, x_4) \wedge (y, f, x_3).\end{aligned}$$

We illustrate in Figure 2 the chased solution for I under $(\mathcal{R}, \Sigma, \mathcal{M}'_{st}, \mathcal{M}_t)$. Notice that there is no solution that has N_1 and N_2 on the same path from c_1 to c_2. Such a condition is desirable for a flight from c_1 to c_2 whose passengers have stopped in both hotels hy and hx, situated in the cities N_1 and N_2, respectively. We finally point out that we cannot capture solutions satisfying this kind of constraints for flights with an arbitrary number of stops without using the Kleene star (as in Example 2.2). ▫

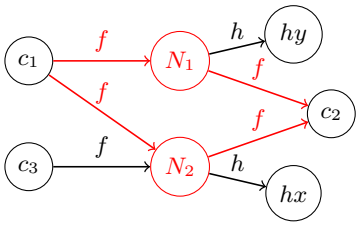


Figure 2: Solution from Example 3.1.

3.2 Graph data exchange

If we consider s-t tgds only, the existence of solutions and query answering can be solved using techniques from graph-to-graph data exchange [5]. In particular, solutions always exist and all solutions are captured by universal representatives defined as graph patterns.

Graph patterns. Let $\mathcal{N}$ be a countably infinite set of labeled null values. A *graph pattern* π over a finite alphabet Σ is a pair (N, D), where $N \subseteq \mathcal{V} \cup \mathcal{N}$ is a finite set of node ids or null values, and $D \subseteq N \times NRE(\Sigma) \times N$, where $NRE(\Sigma)$ denotes the set of all NREs over Σ. The semantics of graph patterns are defined in terms of homomorphisms [4]. Given a graph pattern $\pi = (N, D)$ and a graph database $G = (V, E)$, a *homomorphism* from π into G is a total function $h : N \to V$ s.t.:

1. h is the identity over $N \cap \mathcal{V}$ (i.e., over the node ids from N),
2. for all edges $(u, r, v) \in D$ $(u, v \in N, r \in NRE(\Sigma))$, it holds that $(h(u), h(v)) \in [\![r]\!]_G$.

We write $\pi \to G$ if there exists a homomorphism from π to G. The set of all graphs represented by π over Σ, denoted $Rep_\Sigma(\pi)$ is the set of all graphs G over Σ such that $\pi \to G$.

Universal representatives. Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \varnothing)$ and an instance I of $\mathcal{R}$, a graph pattern π is a *universal representative* of I under Ω if $Sol_\Omega(I) = Rep_\Sigma(\pi)$ [5]. In graph data exchange, universal representatives are computed using an adaptation of the standard *chase* procedure from relational data exchange [11]. Moreover, the variant of chase that is tailored for graph data exchange [5] can be easily adapted to construct a universal representative in our relational-to-graph heterogeneous setting. We illustrate a result of this procedure in Example 3.2. Then, query answering reduces to querying the graph pattern [4] chased as universal representative.

Example 3.2 Take the schemas $\mathcal{R}$ and Σ, the instance I, and the s-t tgds $\mathcal{M}_{st}$ from Example 2.2. The graph pattern π in Figure 3 is a universal representative of all solutions for I under $(\mathcal{R}, \Sigma, \mathcal{M}_{st}, \varnothing)$ i.e., all graphs to which there exists a homomorphism from π are solutions. □

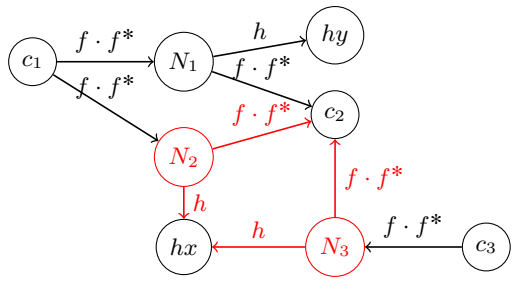


Figure 3: Graph pattern from Example 3.2.

However, notice that the sole s-t tgds might not capture interesting mapping scenarios involving graphs. As an example, the target constraint "a hotel is situated in exactly one city" cannot be expressed in settings such as the one presented in Example 3.2.

4. COMPLEXITY RESULTS

In this section, we present our main contributions. More precisely, we study the complexity of the two problems of interest, namely existence of solutions and query answering, for settings that exhibit target egds (Section 4.1) or target tgds (Section 4.2), respectively.

4.1 Complexity of target egds

First, let us show the intractability of the existence of solutions when we allow egds to our setting.

Theorem 4.1 *Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ where $\mathcal{M}_t$ consists of egds, and an instance I of $\mathcal{R}$, deciding whether there exists a solution for I under Ω is NP-hard.*

Proof. We prove by reduction from 3SAT, known as an NP-complete problem. The reduction works as follows. Given a formula $\rho = C_1 \wedge \ldots \wedge C_k$ in 3CNF over the set of variables $\{x_1, \ldots, x_n\}$, we construct

– The setting $\Omega_\rho = (\mathcal{R}_\rho, \Sigma_\rho, \mathcal{M}_{\rho_{st}}, \mathcal{M}_{\rho_t})$ s.t.

- $\mathcal{R}_\rho = \{R_1, R_2\}$, both unary relations,
- $\Sigma_\rho = \{a, t_1, f_1, \ldots, t_n, f_n\}$.
- $\mathcal{M}_{\rho_{st}}$ contains a unique s-t tgd $R_1(x) \wedge R_2(y) \to (x, a, y) \wedge (x, t_1 + f_1, x) \wedge \ldots \wedge (x, t_n + f_n, x)$.
- $\mathcal{M}_{\rho_t}$ contains two types of egds:

  (*) $(x, (t_j \cdot f_j \cdot a), y) \to (x = y)$, for $1 \leqslant j \leqslant n$,

  (**) $(x, (b_{i_1} \cdot b_{i_2} \cdot b_{i_3} \cdot a), y) \to (x = y)$, for $1 \leqslant i \leqslant k$, for $1 \leqslant i_1, i_2, i_3 \leqslant n$, $x_{i_1}, x_{i_2}, x_{i_3}$ are the variables used in clause C_i, and for $1 \leqslant l \leqslant 3$, b_{i_l} is t_{i_l} if x_{i_l} appears in a negative literal in C_i, and f_{i_l}, otherwise.

– The instance $I_\rho = \{R_1(c_1), R_2(c_2)\}$.

Intuitively, the egds are defined such that a graph collapses if each variable has more than one valuation (*) or the valuation of the variables makes the formula false (**).

We illustrate the construction on the formula $\rho_0 = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_3 \vee \neg x_4)$. We have the s-t tgd $R_1(x) \wedge R_2(y) \to (x, a, y) \wedge (x, (t_1 + f_1), x) \wedge \ldots \wedge (x, (t_4 + f_4), x)$, the egds of type (*) $(x, (t_i \cdot f_i \cdot a), y) \to (x = y)$ (with $1 \leqslant i \leqslant 4$), and the egds of type (**) $(x, (f_1 \cdot t_2 \cdot f_3 \cdot a), y) \to (x = y)$ and $(x, (t_1 \cdot f_3 \cdot t_4 \cdot a), y) \to (x = y)$. Then, the graph in Figure 4 is a solution that encodes the valuation v s.t. $v(x_1) = v(x_2)$=true and $v(x_3) = v(x_4)$=false that makes ρ_0 true.

We claim that there exists a solution for I_ρ under Ω_ρ iff $\rho \in$3SAT.

For the *if* part, we show that the existence of a valuation making ρ true implies the existence of a solution. Take a valuation $v : \{x_1, \ldots, x_n\} \to \{\text{true, false}\}$ making ρ true. Then, construct the graph $G = (\{c_1, c_2\}, E)$ s.t. $E = \{(c_1, a, c_2)\} \cup \{(c_1, t_i, c_1) \mid 1 \leqslant i \leqslant n \text{ and } v(x_i) = \text{true}\} \cup \{(c_1, f_i, c_1) \mid 1 \leqslant i \leqslant n \text{ and } v(x_i) = \text{false}\}$. Note that G and I_ρ satisfy the s-t tgd. Since there is exactly one edge labeled $b_i \in \{t_i, f_i\}$ from c_1 to c_2, the egds of type (*) are satisfied. Moreover, since the b_i's correspond to a valuation making ρ true, there is at least one satisfied literal in every clause of ρ, hence the egds of type (**) are also satisfied. Thus, G is a solution.

For the *only if* part, take a solution G. Since G satisfies the s-t tgd, we infer that G encodes at least one valuation

of every variable. Since G satisfies the egds of type (*), we infer that G encodes at most one valuation of every variable. Thus, G encodes exactly one valuation of every variable. Since G satisfies the egds of type (**), we conclude that G encodes a valuation making ρ true. ▫

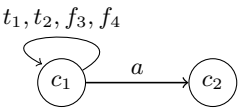


Figure 4: Solution for ρ_0.

We point out that Theorem 4.1 holds even under significantly restricted assumptions that have been used in the proof: (i) *fixed source schema* consisting of two unary relations only, (ii) *fixed source instance*, (iii) s-t tgds using only conjunctions of NREs of the form a or $a + b$ (with $a, b \in \Sigma$) that is a slight relaxation of the restriction from Section 3.1, and (iv) egds that use only NREs of the form $a_1 \cdot \ldots \cdot a_n$, with pairwise distinct $a_1, \ldots, a_n \in \Sigma$ (NREs referred to as "SORE(·)" [2]). Next, we prove that query answering is intractable under the same assumptions and for queries consisting of NREs that use disjunction and concatenation only.

Corollary 4.2 *Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ where $\mathcal{M}_t$ consists of egds, an instance I of $\mathcal{R}$, a NRE r, and a tuple of constants (c_1, c_2), deciding whether $(c_1, c_2) \in cert_\Omega(r, I)$ is coNP-hard.*

Proof. We take the proof of Theorem 4.1, and we additionally consider the NRE $r_\rho = a \cdot a$. We claim that $(c_1, c_2) \in cert_{\Omega_\rho}(r_\rho, I_\rho)$ iff $\rho \notin$3SAT. For the *if* part, notice that $\rho \notin$3SAT implies that there is no solution hence (c_1, c_2) is a certain answer. For the *only if* part, since (c_1, c_2) is a certain answer, we infer that either (i) there is no solution or (ii) there is at least a solution and (c_1, c_2) is an answer for all solutions. But (ii) is false since there exist solutions for which (c_1, c_2) is not an answer for r_ρ (e.g., in Figure 4). Both parts follow directly from the proof of Theorem 4.1. ▫

Finally, we point out that in our reduction the source schema and instance are fixed while the target schema and the mappings are part of the input. Hence, our hardness results stand in terms of *query complexity*. Similar intractability results in the presence of target constraints (particularly in combined complexity) have been shown for relational and XML data exchange [12, 3, 1, 10]. However, our contribution is novel since to the best of our knowledge target constraints on a graph target schema have not been previously considered in the literature, and moreover, we use a fixed source schema and instance in the proof. We also point out that our results are not specific to the relational-to-graph setting and hold in any setting where the target is a graph.

4.2 Complexity of target tgds

In this section, we use *sameAs* constraints instead of egds. First, let us show that the existence of solutions becomes trivial. More precisely, a solution can be computed as follows: (i) chase a graph pattern π using the s-t tgds only, (ii) take a graph G s.t. $\pi \to G$, and (iii) add in G the necessary *sameAs* edges to satisfy the *sameAs* constraints. Recall that the difficulty of deciding the existence of solutions in the case of egds was that we cannot merge two constants. Notice that this difficulty is overcome since we can add *sameAs* edges between any two nodes, even between two constants.

Next, we prove that in the presence of *sameAs* constraints the problem of certain answers is intractable under the same assumptions as in Section 4.1.

Proposition 4.3 *Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ where $\mathcal{M}_t$ consists of sameAs constraints, an instance I of $\mathcal{R}$, a NRE r, and a tuple of constants (c_1, c_2), deciding whether $(c_1, c_2) \in cert_\Omega(r, I)$ is coNP-hard.*

Proof. We take from the proof of Theorem 4.1 the same $\mathcal{R}_\rho$, I_ρ, Σ_ρ, $\mathcal{M}_{\rho_{st}}$, and we replace each $(x = y)$ from $\mathcal{M}_{\rho_t}$ by $(x, sameAs, y)$ to obtain the set of *sameAs* constraints $\mathcal{M}'_{\rho_t}$ and $\Omega'_\rho = (\mathcal{R}_\rho, \Sigma_\rho \cup \{sameAs\}, \mathcal{M}_{\rho_{st}}, \mathcal{M}'_{\rho_t})$. Then, take $r'_\rho = sameAs$. We claim that $(c_1, c_2) \in cert_{\Omega'_\rho}(r'_\rho, I_\rho)$ iff $\rho \notin$3SAT, which follows similarly to Theorem 4.1. ▫

Moreover, we observe that *sameAs* constraints are a particular case of target tgds, and therefore, query answering is intractable in the presence of target tgds.

Corollary 4.4 *Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ where $\mathcal{M}_t$ consists of target tgds, an instance I of $\mathcal{R}$, a NRE r, and a tuple of constants (c_1, c_2), deciding whether $(c_1, c_2) \in cert_\Omega(r, I)$ is coNP-hard.*

5. TOWARDS UNIVERSAL SOLUTIONS

Next, we study a natural adaptation of the standard *chase* procedure [11] to take into account egds. The result of our adapted chase is a graph pattern π. To this purpose, we consider two types of chase steps: (1) for s-t tgds we do similarly to [5] when computing universal representatives in graph data exchange without target constraints, and (2) for egds, for each $\psi_\Sigma(\overline{x}) \to (x_1 = x_2)$, (i) if the images in π of both x_1 and x_2 are constants, then the chase fails, (ii) if one has as image in π a constant and the other a labeled null, then the chase replaces in π the labeled null by the constant, and (iii) if both have labeled nulls as images in π, the chase chooses one of them and replaces it in π with the other.

Example 5.1 For the setting $(\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ and the instance I from Example 2.2, by applying the aforementioned adapted chase we obtain the graph pattern in Figure 5. ▫

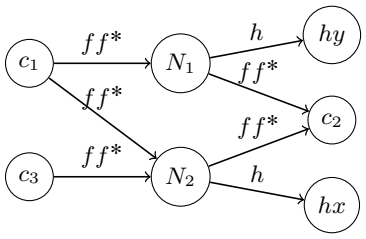


Figure 5: Graph pattern from Example 5.1.

As for relational data exchange, if the chase fails, then there is no solution. As opposed to relational data exchange, we observe that a successful chase does not guarantee the existence of a solution. Intuitively, the difficulty comes from the fact that the chase result is a graph pattern with NREs on the edges (unlike a graph with symbols on the edges). Consequently, there might not exist any graph G s.t. $\pi \to G$ and

G satisfies the target constraints because it may be the case that there is no path satisfying the NREs and the egds at the same time. The following example shows such a situation.

Example 5.2 Take the source schema $\{R, P\}$, an instance $R(c_1)$ and $P(c_2)$, the target schema $\{a, b, c\}$, the s-t tgd $R(x) \wedge P(y) \rightarrow (x, a \cdot (b^* + c^*) \cdot a, y)$, and the egd $(x, a + b + c, y) \rightarrow (x = y)$. The aforementioned adapted chase succeeds and returns the graph pattern π in Figure 6(a). Although the chase has not failed, no solution exists because there is no graph G s.t. $\pi \rightarrow G$ and G satisfies the egds. In particular, the graph G (s.t. $\pi \rightarrow G$) from Figure 6(b) satisfies the s-t tgd but if we try to transform it in a solution we fail because we attempt to merge two constants. ▫

(a) Graph pattern π. (b) Graph G.

Figure 6: Result of a successful chase.

We next show that, even when solutions exist, graph patterns as such cannot be used as universal representatives in the presence of egds.

Proposition 5.3 *Given a setting $\Omega = (\mathcal{R}, \Sigma, \mathcal{M}_{st}, \mathcal{M}_t)$ where $\mathcal{M}_t$ consists of a non-empty set of egds, and an instance I of $\mathcal{R}$, there does not exist a graph pattern π s.t. $Sol_\Omega(I) = Rep_\Sigma(\pi)$.*

Intuitively, let us assume that there exists a graph pattern π s.t. $Sol_\Omega(I) = Rep_\Sigma(\pi)$. Then, if we take a graph $G \in Sol_\Omega$ and a homomorphism $h : \pi \rightarrow G$, we can construct the graph G' by adding nodes and edges to G s.t. some egd is no longer satisfied, thus G' is not a solution for I under Ω, but $h : \pi \rightarrow G'$ is still a homomorphism. The next example clarifies when such a situation can occur.

Example 5.4 The graph in Figure 7 is not a solution for the mappings and instance from Example 2.2 although there exists a homomorphism from the chased graph pattern from Figure 5. ▫

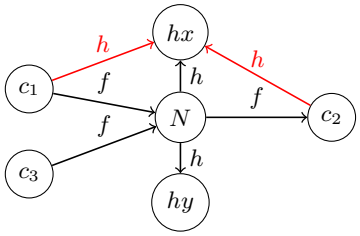


Figure 7: Graph from Example 5.4.

To address the problem of universal representatives in settings involving egds, a natural approach is to define the universal representative as a pair (graph pattern, set of egds). In this case, the solutions are the graphs satisfying the egds and s.t. there exists a homomorphism from the pattern. For example, the universal representatives for Example 2.2 would be the pattern in Figure 5 together with the egd in $\mathcal{M}_t$ from Example 2.2. We also point out that the above discussion can be easily generalized for *sameAs* constraints or arbitrary target tgds.

6. CONCLUSIONS AND FUTURE WORK

We have presented our work on relational-to-graph data exchange. Our main results are the proofs of intractability of the existence of solutions and query answering that hold even under considerable restrictions of the problem setting. As future work, we would like to investigate the complexity upper bounds of these problems and look for tractable fragments to have a complete picture of the difficulty of our setting. A natural question that remains open is how to query universal representatives consisting of a pair (graph pattern, set of target constraints). We would also like to investigate practical scenarios of relational-to-RDF data exchange and other classes of heterogeneous schema mappings. Additionally, it would be interesting to combine existing learning techniques for relational [9] and graph [8] queries in order to propose algorithms that automatically infer relational-to-graph mappings from examples provided by the user.

7. REFERENCES

[1] S. Amano, C. David, L. Libkin, and F. Murlak. XML schema mappings: Data exchange and metadata management. *J. ACM*, 61(2):12, 2014.

[2] T. Antonopoulos, F. Neven, and F. Servais. Definability problems for graph query languages. In *ICDT*, pages 141–152, 2013.

[3] M. Arenas and L. Libkin. XML data exchange: Consistency and query answering. *J. ACM*, 55(2), 2008.

[4] P. Barceló, L. Libkin, and J. L. Reutter. Querying graph patterns. In *PODS*, pages 199–210, 2011.

[5] P. Barceló, J. Pérez, and J. L. Reutter. Schema mappings and data exchange for graph databases. In *ICDT*, pages 189–200, 2013.

[6] C. Beeri and M. Y. Vardi. A proof procedure for data dependencies. *J. ACM*, 31(4):718–741, 1984.

[7] Z. Bellahsene, A. Bonifati, and E. Rahm, editors. *Schema Matching and Mapping*. Springer, 2011.

[8] A. Bonifati, R. Ciucanu, and A. Lemay. Learning path queries on graph databases. In *EDBT*, 2015.

[9] A. Bonifati, R. Ciucanu, and S. Staworko. Interactive inference of join queries. In *EDBT*, pages 451–462, 2014.

[10] A. Calì, G. Gottlob, and M. Kifer. Taming the infinite chase: Query answering under expressive relational constraints. *J. Artif. Intell. Res. (JAIR)*, 48:115–174, 2013.

[11] R. Fagin, P. G. Kolaitis, R. J. Miller, and L. Popa. Data exchange: semantics and query answering. *Theor. Comput. Sci.*, 336(1):89–124, 2005.

[12] P. G. Kolaitis, J. Panttaja, and W. C. Tan. The complexity of data exchange. In *PODS*, pages 30–39, 2006.

[13] P. G. Kolaitis, R. Pichler, E. Sallinger, and V. Savenkov. Nested dependencies: structure and reasoning. In *PODS*, pages 176–187, 2014.

[14] A. Poggi, D. Lembo, D. Calvanese, G. De Giacomo, M. Lenzerini, and R. Rosati. Linking data to ontologies. *J. Data Semantics*, 10:133–173, 2008.

[15] L. Popa, Y. Velegrakis, R. J. Miller, M. A. Hernández, and R. Fagin. Translating web data. In *VLDB*, pages 598–609, 2002.

[16] J. Sequeda, M. Arenas, and D. P. Miranker. On directly mapping relational databases to RDF and OWL. In *WWW*, pages 649–658, 2012.